논문 2009-46SD-10-9

구조적 LDPC 부호의 저복잡도 및 고속 부호화기 설계

(Design of Low Complexity and High Throughput Encoder for Structured LDPC Codes)

정 용 민*, 정 윤 호**, 김 재 석***

(Yongmin Jung, Yunho Jung, and Jaeseok Kim)

요 약

본 논문은 저 복잡도와 높은 throughput을 지원하는 LDPC 부호화기의 구조에 대하여 제안한다. LDPC 부호화기가 갖는 높은 복잡도 문제를 해결하기 위하여 기존의 복잡도가 높은 행렬 곱셈 연산기 대신에 간소화된 행렬 곱셈 연산기가 제안되었다. 또한 높은 throughput을 지원하기 위하여 행렬 곱셈 연산시 행 방향 연산 및 부분 병렬처리 연산을 적용하였다. 제안된 부호화기 구조의 로직 게이트와 메모리 사용량은 기존의 5단 파이프라인 부호화기의 구조에 비하여 각각 37.4%와 56.7%씩 감소하였다. 또한 40MHz 클럭 주파수에 대해 기존의 부호화기에 비하여 3배 이상의 throughput인 최대 800Mbps의 throughput을 지원한다.

Abstract

This paper presents the design results of a low complexity and high throughput LDPC encoder structure. In order to solve the high complexity problem of the LDPC encoder, a simplified matrix-vector multiplier is proposed instead of the conventional complex matrix-vector multiplier. The proposed encoder also adopts a partially parallel structure and performs column-wise operations in matrix-vector multiplication to achieve high throughput. Implementation results show that the proposed architecture reduces the number of logic gates and memory elements by 37.4% and 56.7%, compared with existing five-stage pipelined architecture. The proposed encoder also supports 800Mbps throughput at 40MHz clock frequency which is improved about three times more than the existing architecture.

Keywords : Column-wise multiplication, LDPC code, simplified matrix-vector multiplication

Ⅰ. Introduction

Recently low-density parity-check (LDPC) codes have received tremendous attention in wireless communication systems due to its excellent error correction capability close to the Shannon's channel capacity for the AWGN channel[1]. LDPC codes have been selected by various wireless communication systems such as IEEE 802.11n wireless LAN (WLAN) and IEEE 802.16e mobile WiMAX. However, a high encoding complexity is a major drawback of LDPC codes in spite of its excellence performance[2~7].

Since LDPC code is a class of linear block codes, most of the encoding operations are matrix-vector multiplications. The encoding complexity of straightforward method is quadratically proportional to a codeword block length which leads to the high encoder complexity. The block codes also require

* 학생회원, *** 정회원, 연세대학교 전기전자공학과
(Department of Electrical and Electronic Engineering, Yonsei University)
** 평생회원, 한국항공대학교
(Korea Aerospace University)
※ 본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT 연구센터 지원사업의 연구결과로 수행되었음. (IITA-2009-(C1090-0902-0012))
접수일자: 2009년7월2일, 수정완료일: 2009년9월22일

many operations to generate parity bits, and therefore it causes high latency and low throughput of encoder. However, since the communication systems which employ LDPC codes as a forward error correction (FEC) support a high data rate, the architecture design of a high throughput LDPC encoder is essential. Therefore, the low complexity and the high throughput are very significant factors in LDPC encoder design.

In order to obtain the linear complexity encoder, Richardson and Urbanke introduced an encoding scheme with the modified parity check matrix[5]. The linear encoding scheme transforms the random parity check matrix into the systematic matrix by row or column permutations. The vector additions are able to displace the matrix-vector multiplications due to the systematic feature of the refined matrix. Since this encoding scheme reduces the encoder complexity for any parity check matrix, it has been mostly used for the design of LDPC encoder. However, this encoding scheme requires many matrix-vector multiplications, which cause a lot of memory requirement saving results in pipelined structure[1, 6]. For linear complexity encoder, extended irregular repeat accumulate (EIRA) codes have been proposed which need fewer matrix-vector multiplications than Richardson's encoding scheme. In EIRA codes, the parity check matrix is partitioned into two sub-matrices. One of them is represented as a dual diagonal matrix which replaces matrix-vector multipliers by the vector adders, and the low complexity encoder thus is achievable[8~10].

Although the above linear encoding schemes can reduce the encoder complexity, they cannot decrease memory that saves thousands of parity check matrix. Quasi-cyclic (QC) LDPC codes have been proposed to reduce the memory size. In QC LDPC codes, the parity check matrix is defined as an array of sub-matrices that are square matrices composed of either an identity matrices with column permutation or a zero matrices. This properties make it possible to reduce the memory usage dramatically[11~12]. Consequently, it is recommended to use QC and EIRA codes for very low complexity LDPC encoder.

Both the complexity and the throughput are affected by the encoder architecture even if the same linear encoding scheme is heme. In order to achieve low complexity and h oh throughput, block LDPC encoder based on Richardson's encoding scheme is proposed by Zhong[1]. The encoder adopted low complexity matrix-vector multiplier with simple hardwongd interconnection network and the partially parallel processing with task schedulingow complexiмemory ele[1].scheme ptethe matrix-vector multiplier and the strict condition is heask schedulingcodetrict the peris LDPce improve[1].. To obeaicodire effective architecture icocomplexity and throughput, this paper proposes QC EIRA based LDPC encoder architecture. The proposed architecture uses simplified gow complexiмemory ele[1].sc without memory ele[1].scto reduce the complexityow oh throughputhiteclso achieved by using the column-wise mory ele[1].sc and partially parallel processing architecture.

The remainder of this paper is organized as follows. Section Ⅱ covers QC-EIRA code as the linear encoding scheme. Section Ⅲ proposes the low complexity and high throughput LDPC encoder architecture. Section Ⅳ shows the implementation results. Conclusions are given in Section Ⅴ.

Ⅱ. QC-EIRA LDPC Encoding Scheme

1. Structural Property of QC-LDPC Codes

In QC-LDPC codes, the parity check matrix **H** consists of array of sub-matrices as follows,

$$\boldsymbol{H}=\begin{pmatrix} h_{0,0} & \cdots & h_{0,j} & h_{0,l-1} \\ \vdots & \ddots & \ddots & \vdots \\ h_{i,0} & \cdots & h_{i,j} & h_{i,l-1} \\ h_{p-1,0} & \cdots & h_{p-1,j} & h_{p-1,l-1} \end{pmatrix}, \quad (1)$$

where each sub-matrix $h_{i,j}$ is $p \times p$ square matrix. The square matrices are either zero matrices or identity matrices with column permutation.

A random parity check matrix needs a large number of memory elements to save the parity check matrix. It also needs a general matrix-vector multiplier occupying a large area in LDPC encoder. However, QC-LDPC codes fo the structured LDPC codes not only reduce the memory requirement remarkably but also replace the matrix-vector multiplier by a simple logic element like a cyclic shifter. Accordingly, the structured LDPC codes are employed as FEC block in several wireless communication systems[2~3].

2. EIRA Linear Encoding Scheme

In EIRA codes, the parity check matrix is partitioned into the two sub-matrices as follows,

$$\boldsymbol{H} = [\boldsymbol{H_1}\ \boldsymbol{H_2}], \tag{2}$$

which is a sparse $(np-kp)\times np$ matrix. $\boldsymbol{H_1}$ and $\boldsymbol{H_2}$ represents $(np-kp)\times kp$ matrix and $mp\times mp$ matrix. n, k and m stand for a number of sub-blocks of the codeword, information and parity bits, respectively. Lastly, p denotes the size of each sub-block. When the parity check matrix is described as (2), parity bits are obtained as follows,

$$\boldsymbol{c} = [\boldsymbol{I}\ \boldsymbol{P}], \tag{3}$$

$$\boldsymbol{H}\boldsymbol{c}^T = 0, \tag{4}$$

$$\boldsymbol{H_1}\boldsymbol{I}^T + \boldsymbol{H_2}\boldsymbol{P}^T = 0, \tag{5}$$

$$\boldsymbol{P}^T = \boldsymbol{H_2}^{-1}\boldsymbol{H_1}\boldsymbol{I}^T, \tag{6}$$

where $\boldsymbol{c}$, $\boldsymbol{I}$ and $\boldsymbol{P}$ represent codeword bits, information bits and parity bits, respectively. Generally, the matrix inversion changes the sparse matrix $\boldsymbol{H_2}$ into much complex matrix. $\boldsymbol{H_2}$ in EIRA code has the systematic form as (7)[8].

$$\boldsymbol{H_2} = \begin{bmatrix} h_{2(0,0)} & 0 & & \\ \vdots & 0 & 0 & -\mathbf{1} \\ h_{2(i,0)} & & \ddots\ \ddots & \\ \vdots & -\mathbf{1} & 0 & 0 \\ h_{2(m-1,0)} & & & 0 \end{bmatrix}. \tag{7}$$

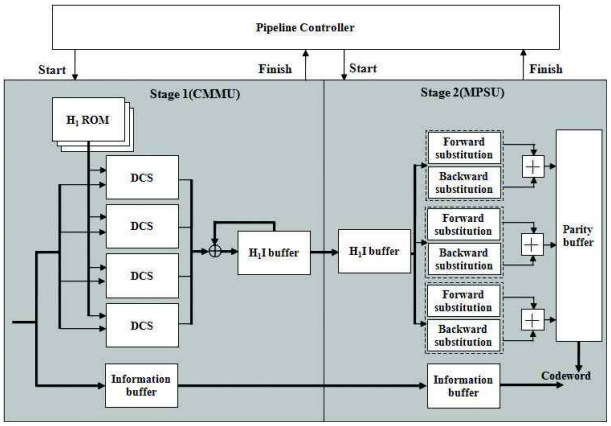

그림 1. 제안된 2단 LDPC 부호화기의 구조
Fig. 1. Overall architecture of proposed two-stage LDPC encoder.

Digit 0 and -1 in (7) stand for $p\times p$ identity matrix and zero matrix, respectively. $h_{2(i,0)}$ represents the permutation values of the first column sub-matrices. The systematic form of (7) is able to simplify the computation of (6).

Ⅲ. Proposed LDPC Encoder Architecture

1. Overall Architecture of Proposed LDPC Encoder

Fig. 1 shows the overall architecture of the proposed LDPC encoder which is pipelined in two-stage. Column-wise matrix multiplication unit (CMMU) computes $\boldsymbol{H_1}\boldsymbol{I}^T$ in the first stage and multiple parity substitution unit (MPSU) generates parity bits in the second stage. CMMU uses decomposed cyclic shifter (DCS) as a matrix-vector multiplier instead of the conventional matrix-vector multiplier using the memory elements[1] or the cyclic shifter like the logarithmic shifter[9]. DCS multiplies the information bits by each row of the matrix $\boldsymbol{H_1}$ saved in $\boldsymbol{H_1}$ ROM. The results of CMMU are saved in $\boldsymbol{H_1}\boldsymbol{I}$ buffer. MPSU multiplies the inverse of $\boldsymbol{H_2}$ by the results of CMMU to compute parity bits based on (6). Since the latency of CMMU is efficiently reduced by using the column-wise matrix multiplication, the throughput of LDPC encoder is determined by MPSU. In order to raise the

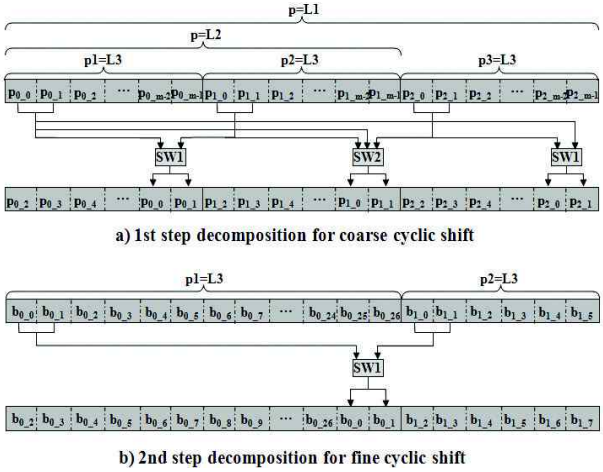

그림 2. 구조적 LDPC 부호화기의 decomposed cyclic shift.

Fig. 2. Decomposed cyclic shift for structured LDPC codes.

throughput, MPSU uses the parallel processing architecture as Fig. 1. The results of MPSU are saved in the parity buffer. The controller in Fig. 1 generates a start signal to operate each stage operate, and then receives a finish signal after each block finishes the operation.

2. Column-wise Matrix-vector Multiplication Unit

Most of the LDPC encoding operations are the matrix-vector multiplications such as $\boldsymbol{H_1 I}^T$ in (6) that cause a large hardware complexity and a high latency. DCS and column-wise matrix multiplication are applied to CMMU in the proposed encoder to reduce the hardware overhead and latency of $\boldsymbol{H_1 I}^T$.

A. The Structure of DCS

The parity check matrix of the structured LDPC codes is divided into the identity matrices with permutation and zero matrices. This means that the matrix-vector multiplication can be replaced with cyclic shifter and the logarithmic shifter is generally used as the cyclic shifter due to low complexity. However, since the logarithmic shifter cannot support the permutation for the different size sequences, the different size logarithmic shifter is required to support various codeword block lengths. Even though memory based matrix-vector multiplier supports different size permutation with memory elements and address generators, it causes very high complexity. In this paper, we propose a very efficient cyclic shifter called DCS as shown in Fig. 2.

Suppose that LDPC code supports three codeword block lengths with different parity check matrices. Each parity check matrix is partitioned into the square sub-matrices with size of L1, L2 or L3 as in Fig. 2. Also, an information block can be divided into sub-blocks in the same manner. DCS multiplies the sub-matrix by sub-block of information and divides the sub-block into several tiny sub-blocks, p_{i_j}, at step 1 depicted in Fig. 2. The unit of cyclic shift at step 1 is equal to the size of p_{i_j} which is not one bit. It is effective for complexity of DCS when the size of tiny sub-block is the same as that of common divisor of L1, L2 and L3. The cyclic shift processing at step 1 is similar to the logarithmic shifter. However, the 1st step cannot complete the cyclic shift because the unit of shift is not one bit. After a coarse cyclic shift, a fine cyclic shift is performed at the 2nd step depicted in Fig. 2. The resolution of cyclic shift at step 2 is one bit described as p_{i_j}. In order to support different size of cyclic shift such as L1, L2 and L3, switches (SW) select different bits as the length of sub-block. For example, the first switch, SW1, selects p_{i_j} when the sub-block length is L1. In other cases, SW1 chooses p_{i_j} at step 1. Consequently, decomposing cyclic shifter into two steps occupies less areas than the logarithmic shifter and another matrix-vector multiplier using memory elements.

B. Column-wise Matrix-vector Multiplication

The throughput of LDPC encoder is determined by a latency of matrix-vector multiplier. Hence, the latency of CMMU multiplying parity check matrix by information bits should be reduced to increase the throughput. In order to reduce latency, CMMU computes $\boldsymbol{H_1 I}^T$ of (6) in column-wise manner.

LDPC encoder receives a single input bit or multiple input bits per single clock cycle. CMMU computes the matrix-vector multiplication for already received bits during the period that current input bits are coming. For the parity check matrix in (1), $\boldsymbol{H_1 I}^T$ is computed with column-wise fashion as follows,

$$\begin{aligned} q_0^j &= q_0^{j-1} + h_{0,j} \cdot I_j \\ q_1^j &= q_1^{j-1} + h_{1,j} \cdot I_j \\ &\vdots \\ q_{p-1}^j &= q_{p-1}^{j-1} + h_{p-1,j} \cdot I_j \end{aligned} \tag{8}$$

where I_j denotes the jth information sub-block and $h_{i,j}$ stands for the square sub-matrix of the parity check matrix at the i_{th} row and the j_{th} column. Lastly, q_i^j represents the j_{th} computation results for the i_{th} row of the parity check matrix. When the j_{th} information sub-block is entered in CMMU, it is multiplied by sub-matrices every row and the j_{th} column of $\boldsymbol{H_1}$. The results are added to q_i^{j-1}, and then q_i^j are stored in the $\boldsymbol{H_1 I}$ buffer. For other information sub-blocks, CMMU computes matrix-vector multiplication in the same pattern. The final value of $\boldsymbol{q} = [q_0\ q_1 \cdots\ q_{p-1}]^T$ are stored in buffer. Since CMMU computes multiplication during the interval when input bits are entering, a low latency computing of $\boldsymbol{H_1 I}^T$ is achievable.

A parallel processing architecture is applied to obtain less latency at CMMU. However, since a fully parallel structure of cyclic shifter causes a large hardware complexity, partially parallel processing structure during the time interval between two information sub-blocks is applied by considering the tradeoff between latency and complexity. Assume that the size of sub-matrix h_{ij} is 40 and multiple input bits per each clock cycle is 8. Since the size of sub-matrix is 40, CMMU waits information bits for five clock cycles. Therefore, it is possible to computes multiplication in partially parallel architecture as shown in Fig. 3. The number of cyclic shifters in partially parallel

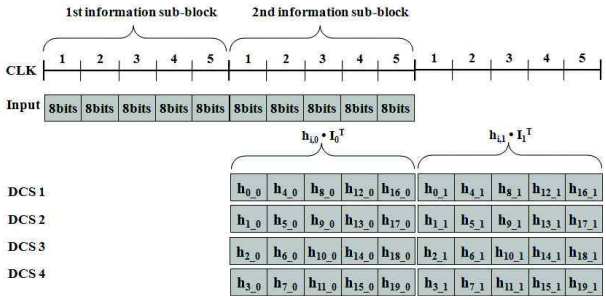

그림 3. 열 방향 행렬 곱셈 연산의 부분 병렬 처리
Fig. 3. Partially parallel processing in CMMU.

architecture is given by,

$$a = \lfloor p/IN \rfloor, \tag{9}$$

$$b = \lceil Row/a \rceil, \tag{10}$$

where p, IN and Row stand for the size of sub-matrix, the number of input bits per clock cycle, and the number of rows in (1), respectively. $\lfloor x \rfloor$ denotes the largest integer less than or equal to x and $\lceil x \rceil$ denotes the smallest integer greater than or equal to x. Finally, the value of N in (10) represents the number of cyclic shifter in CMMU.

3. Multiple Parity Substitution Unit (MPSU)

MPSU generates parity bits by multiplying the results of the CMMU by $\boldsymbol{H_2}^{-1}$as in (6). The inverse matrix $\boldsymbol{H_2}^{-1}$ has the systematic form by EIRA property, so that the multiplication is replaced with the addition by using the substitution operation. In MPSU, double direction substitutions, a forward and backward substitution, are applied to decrease the latency, which are represented as,

$$\boldsymbol{H_1 I}^T = [q_0\ q_1 \cdots\ q_{m-1}]^T, \tag{11}$$

$$f_i = q_i + q_{i+1}, \qquad 0 \le i \le \frac{m}{2} - 2, \tag{12}$$

$$b_0 = q_{m-1} + q_{m-2}, \tag{13}$$

$$b_i = b_{i-1} + q_{m-2-i}, 1 \le i \le \frac{m}{2} - 3, \tag{14}$$

where q_i means each sub-block of the results of $\boldsymbol{H_1}$

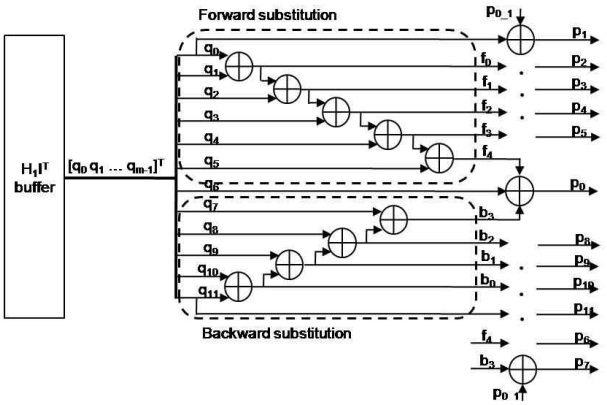

그림 4. 패리티 연산 유닛의 병렬처리 구조
Fig. 4. Parallel structure of MPSU.

$\boldsymbol{I}^T$. Also, f_i and b_i represent the forward and backward substitution results. The substitution results are used to generate parity bits as follows,

$$p_0 = f_{m/2-2} + b_{m/2-3} + q_{m/2} \tag{15}$$

$$p_{0\_1} = c_1 p_0, \tag{16}$$

where p_0, the summation of $\boldsymbol{H_1 I}^T$, means the first sub-block of the parity bits and p_{0_1} represents one step cyclic shift. The rest of the parity bits are obtained as follows,

$$\begin{aligned} p_1 &= p_{0\_1} + q_{0,} \\ p_i &= p_{0\_1} + f_{i-2}, \quad 2 \le i \le \frac{m}{2} \\ p_{m-1} &= p_{0\_1} + q_{m-1}, \\ p_j &= p_{0\_1} + b_{m-2-j}, \frac{m}{2} + 1 \le j \le m-2, \\ \boldsymbol{p} &= [p_0\, p_1 \cdots\ p_{m-1}]^T. \end{aligned} \tag{17, 18}$$

Fig. 4 describes the MPSU architecture which requires p clock cycles to generate all parity bits. The partially parallel architecture is also employed in MPSU to reduce the latency and also increase the throughput of the encoder.

Ⅳ. Design and Implementation Results

1. The Hardware Complexity

The proposed encoder which is appropriate for the structured LDPC codes is built up of two-stage pipelined structure. In stead of the conventional cyclic shifter or matrix-vector multiplier, DCS is proposed to reduce the hardware complexity.

Table 1 shows the logic gates of the proposed LDPC encoder, where m, p, N_p, and T denote the number of sub-blocks in parity block, the sub-block size, a number of parallelism and time for hardware resource sharing, respectively. In CMMU, the required number of XOR gates is $p \cdot m/T$. The logic gates of DCS is the summation of cyclic shifting logic $\lceil \log_2 p \rceil \cdot p$ and the switch logic $21.5 \cdot (2^{\lceil \log_2 p \rceil - 2} - 1)$ to support different size shift. The amount of the required XOR logic gate in MPSU to generate parity bits during p/M clock cycles is $2M \cdot (m-1)$.

The memory requirement in terms of bits for each stage is shown in Table 2. n denotes the number of sub-blocks in a codeword block where stages have a memory for information bits. Two memories are required to support pipeline operation in CMMU, so CMMU needs memory of $2 \cdot (p \cdot m)$ bits. MPSU has memory of $2 \cdot (p \cdot m)$ bits to store the results of CMMU and MPSU.

표 1. 제안된 부호화기의 로직 게이트 카운트
Table 1. Logic gate counts of the proposed encoder.

Block	Logic Gate Count (2-input NAND)
CMMU	$3p \cdot m/T$
DCS	$N_p \cdot (\lceil \log_2 p \rceil \cdot p + 21.5 \cdot (2^{\lceil \log_2 p \rceil - 2} - 1))$
MPSU	$6 \cdot 2M \cdot (m-1)$

표 2. 메모리 사용량
Table 2. Required memory size.

Block	CMMU	MPSU
Information	$p \cdot (n-m)$	$p \cdot (n-m)$
DCS	$2(p \cdot m)$	$p \cdot m$
MPSU	-	$p \cdot m$
Total	$p \cdot n + p \cdot m$	$p \cdot n + p \cdot m$

표 3. LDPC 부호화기의 복잡도 비교
Table 3. Complexity comparison of LDPC encoder.

	Five-stage Pipelined Encoder[1]	Proposed Encoder
Logic gate	$(1+\theta)\cdot\|P\|\cdot(3+8\lceil\log_2 p\rceil)+\frac{1}{2}g^2$	$N_p(\lceil\log_2 p\rceil\cdot p+21.5(2^{\lceil\log_2 p\rceil-2}-1))$ $+3p\cdot m/T+6(m-1)$
Total memory	$14p\cdot m-2g$	$2p\cdot(m+n)$

표 4. IEEE 802.11n LDPC 부호화기의 복잡도 및 throughput 비교
Table 4. Performance comparison of IEEE 802.11n LDPC encoder.

	Five-stage pipelined encoder[1]	Proposed encoder	Reduction rate
Logic gate	10.7K	6.7K	37.4%
Total memory	13,400bits	5,800bits	56.7%
Throughput@40MHz	266Mbps	800Mbps	-

Table 3 shows the hardware complexity comparison results between existing five-stage pipelined architecture[1] and the proposed encoder. $|P|$, θ g denote the total number of non-zero blocks in the parity check matrix, the ratio of the number of non-zero blocks in the approximate lower triangular matrix divided by $|P|$ and the size of matrix ϕ[1], respectively. Since the value of $|P|$ is much larger than N_p, the number of logic gates of the proposed encoder is less than that of the five-stage pipelined architecture. The amount of the total memory is also reduced from $14p\cdot m-2g$ to $2p\cdot(m+n)$.

2. The Throughput

The stage in pipelined architecture which has the maximum latency determines the throughput of the LDPC encoder. CMMU computes the matrix-vector multiplication processing with short latency by the column-wise multiplication. It obtains the results within a few clock cycles after all input bits are entered. On the other hand, MPSU necessitates p clock cycles considering tradeoffs between the throughput and the hardware complexity. Therefore, the latency of MPSU is the dominant factor of the decision of the encoder throughput. For the $n-m$ information sub-blocks and the clock frequency f_c, the throughput of LDPC is determined by $\frac{(n-m)\cdot p\cdot f_c}{CPC}$, where clock cycle per codeword (CPC) means the number of clock cycles needed in encoding and is equal to the maximum number of clock period for each stage. Therefore, CPC in the architecture is the same as the clock latency of MPSU, p/M, and the throughput of the encoder is given by,

$$\frac{(n-m)\cdot p\cdot f_c\cdot M}{p}=(n-m)\cdot M\cdot f_c. \quad (19)$$

The throughput of the five-stage pipelined architecture[1], On the other hand, is determined as follows,

$$\frac{(n-m)\cdot f_c}{m}. \quad (20)$$

Since m and M indicate an integer larger than one, the throughput of the proposed encoder is higher than one of the five-stage pipelined architecture.

3. The Performance Comparison in IEEE 802.11n LDPC

Table 4 shows the performance comparison results of the five stage LDPC encoder[1] and the proposed LDPC encoder when applied to IEEE 802.11n WLAN systems. The proposed LDPC encoder reduces the

number of logic gate and the number of total memory by 37.4% and 56.7%, respectively, compared to the five-stage pipelined LDPC encoder. While the five-stage pipelined LDPC encoder supports throughput up to 266Mbps at 40MHz, the maximum throughput of the proposed LDPC encoder reaches 800Mbps at the same clock frequency, which is improved by about three times faster than the existing architecture.

Ⅴ. Conclusions

This paper proposed the low complexity and the high throughput LDPC encoder which is very suitable for high throughput wireless communication systems. The encoder is built up of the two-stage pipelined architecture based on EIRA and structured LDPC codes. At the first pipelined stage, the column-wise multiplication is employed to reduce the latency of the matrix-vector multiplications. At the second stage, the partially parallel processing architecture is applied to obtain high throughput. The proposed LDPC encoder reduces the number of logic gate and memory by 37.4% and 56.7%, respectively, compared with the five-stage pipelined LDPC encoder[1] in IEEE 802.11n WLAN system. The throughput reaches 800Mbps at 40MHz clock frequency, which satisfies the maximum data rate, 600Mbps, of IEEE 802.11n WLAN systems. Therefore, the proposed low complexity LDPC encoder is expected to play an important role in high-speed wireless communication systems.

Reference

[1] H. Zhong, and T. Zhang, “Block-LDPC: a practical LDPC coding system design approach,” IEEE Trans. On Circuits and Systems--I: Regular Papers, vol. 52, no. 4, pp. 766-775, Apr. 2005.

[2] IEEE 802.11nTM/D3.00, “Draft Amendment to STANDARD information Technology Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications,” IEEE 802.11 document.

[3] IEEE Std 802.16eTM-2005, “IEEE Standard for Local and metropolitan area networks Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems,”

[4] S. Y. Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, “On the design of low-density parity-check codes within 0.0045 dB of Shannon limit,” IEEE Communications Letters, vol. 5, pp. 58-60, Feb. 2001.

[5] T. J. Richardson, and R. Urbanke, “Efficient encoding of low-density parity-check codes,” IEEE Trans. Information Theory, vol. 47, pp. 638-656, Feb. 2001.

[6] D. U. Lee, W.Luk, C.Wang, and C. Jones, “A flexible hardware encoder for low-density parity-check codes,” Proceedings of the 12th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, vol. 52, no. 4, pp. 766-775, Apr. 2005.

[7] 김정기, 발라카난, 이문호, “Wibro 시스템을 위한 고속 LDPC 인코더 설계,” 대한전자공학회 논문지 제 45 권 TC 편 제 7호, pp. 1-8, Jul. 2008.

[8] M. Yang, W. E. Ryan, and Y. Li, “Design of efficiently encodable moderate-length high-rate irregular LDPC codes,” IEEE Trans. On Communications, vol. 52, no. 4, pp. 564-571, Apr. 2004.

[9] Y. Sun, M.Karkooti, and J. R. Cavallaro, “High throughput, parallel, scalable LDPC encoder/decoder architecture for OFDM systems,” EEE Dallas/CAS Workshop on Design, Applications, Integration and Software, pp. 39-42, Oct. 2006.

[10] 이찬호, 박재근, “하드웨어 구현에 적합한 효율적인 LDPC 코덱의 설계,” 대한전자공학회 논문지 제 43 권 SD 편 제 7호, pp. 50-57, Jul. 2006.

[11] M. P. C. Fossorier, “Quasi-cyclic low density parity check codes from circulant permutation matrices,” IEEE Trans. Information Theory, vol. 50, pp. 1788-1794, Aug. 2004.

[12] Y. Dai, N. Chen, and Z. Yan, “Memory-Efficient Decoder Architecture for Quasi-Cyclic LDPC Codes,” IEEE Trans. Circuits and Systems--I: Regular Papers, vol. 55, no. 9, pp. 2898-2911, Oct. 2008.

저 자 소 개

정 용 민(학생회원)
2007년 연세대학교 전기전자 공학과 학사
2009년 연세대학교 전기전자 공학과 석사
2009년～현재 연세대학교 전기전자공학과 박사과정
<주관심분야 : MIMO/OFDM 통신시스템, 채널 부호이론, 모뎀 SoC 설계>

김 재 석(정회원)
1977년 연세대학교 전자공학과 학사
1979년 한국과학원 전기 및 전자공학과 석사
1988년 Rensselaer Polytechnic Institute, NY. 박사
1988년～1993년 AT&T Bell Lab. 연구원
1993년～1996년 한국전자통신연구원 책임 연구원
1996년～현재 연세대학교 전기전자공학과 교수
<주관심분야 : 통신 SoC 설계, 고속 멀티미디어 IP 설계>

정 윤 호(종신회원)
1998년 연세대학교 전자공학과 학사
2000년 연세대학교 전기전자 공학과 석사
2005년 연세대학교 전기전자 공학과 박사
2005년～2007년 삼성전자 책임연구원
2007년～2008년 연세대학교 연구교수
2008년～현재 한국항공대학교 조교수
<주관심분야 : MIMO/OFDM 통신시스템, VLSI 신호처리, 모뎀 SoC 설계>